



```
1 import java.util.Scanner;
2
3 import java.io.*;
4
5 public class Main extends AllIn {
6     String fname;
7
8     public Main() {
9         System.out.println("Constructor");
10        getFileName();
11        readFileContents();
12    }
13    public void getFileName()
14    {
15        Scanner in = new Scanner(System.in);
16
17        System.out.println("Enter file name please.");
18        fname = in.nextLine();
19        System.out.println("You entered "+fname);
20    }
21    public void readFileContents()
22    {
23        boolean looping;
24        DataInputStream in;
25        String line;
26        int j, len;
27        char ch;
28
29        /* Read input from file and process. */
30        try {
31            in = new DataInputStream(new FileInputStream(fname));
32
33            looping = true;
34            while(looping) {
35                /* Get a line of input from the file. */
36                if (null == (line = in.readLine())) {
37                    looping = false;
38                    /* Close and free up system resource. */
39                    in.close();
40                }
41                else {
42                    AllIn val = new AllIn();
43                    //Validation
44                    if( line.equals("parentheses tests")){
45                        System.out.println("\n" + line + "\n");
46                        line = in.readLine();
47
48                        while(!line.equals("postfix equation solving")){
49
50                            System.out.println("\n" + line + "\n"); System.out.println((val.Validation(line)));
51
52                            line = in.readLine();
53                            System.out.println("\n");
54                        }
55                    }
56                    // Evaluation
57                    System.out.println("*****");
58                    if( line.equals("postfix equation solving")){
```

```
> sh -c javac -classpath .:target/dependency/ -d . $(find . -type f -name '*.java')
Note: ./Main.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
> java -classpath .:target/dependency/* Main
Hello TV land!
Constructor
Enter file name please.
buff.txt
You entered buff.txt
```

parentheses tests

((1+2)*3

Invalid: odd number of brackets

(1+2))*3

Invalid: odd number of brackets

y=(z*j)/(b*8)+2

Valid: even number of brackets

postfix equation solving

line = 73+9*
= 90

line = 52+19**
= 63

line = 89*35+-
= 64

line = 21+84/*
= 5

infix to postfix conversion

line = (8+4)*9
= 8 4 + 9 *

line = (6+3)*(2*8)
= 6 3 + 2 8 * *

line = (9*8)-(5+3)
= 9 8 * 5 3 + -

line = 3+(9/3)

Files

AllIn.java
buff.txt
charStack.java
intStack.java
Main.java
myQueue.java

CPU
RAM
Storage

```
1  
2 class charStack {  
3     int top;  
4     char[] charStack = new char[100];  
5     char c;  
6  
7     public void init()  
8     {  
9         top = -1;  
10    }  
11  
12    public void push(char item)  
13    {  
14        top++;  
15        charStack[top] = item;  
16    }  
17  
18    public char pop()  
19    {  
20  
21        char c;  
22        c = charStack[top];  
23        top--;  
24        return c;  
25    }  
26  
27    public boolean isEmpty()  
28    {  
29        if(top == -1){  
30            return true;  
31        }  
32        else return false;  
33    }  
34  
35    public int getTop()  
36    {  
37        return charStack[top];  
38    }  
39  
40    public void showStack()  
41    {  
42        int j;  
43        for(j=0;j<=top;j++){  
44            System.out.println(charStack[j]);  
45        }  
46    }  
47 }  
48
```

Console

line = 73+9*
= 90

line = 52+19**
= 63

line = 89*35+-
= 64

line = 21+84/+
= 5

infix to postfix conversion
line = (8+4)*9
= 8 4 + 9 *

line = (6+3)*(2*8)
= 6 3 + 2 8 * *

line = (9*8)-(5*3)
= 9 8 * 5 3 + -

line = 3*(9/3)
= 3 9 3 / +

Parentheses check, infix to postfix conversion, postfix evaluation
line = (8+4)*9
Valid: even number of brackets
8 4 + 9 *
= 108

line = (6+3)*(2*8)
Valid: even number of brackets
6 3 + 2 8 * *
= 144

line = (9*8)-(5*3)
Valid: even number of brackets
9 8 * 5 3 + -
= 64

line = 3*(9/3)
Valid: even number of brackets
3 9 3 / +
= 6

Bye-bye!
> []

Files

AllIn.java

buff.txt

charStack.java

intStack.java

Main.java

myQueue.java

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

symbol = eq.charAt(i);

if(symbol == '[' || symbol == '(' || symbol == '{'){

s.push(symbol);

}

else if(symbol == ']' || symbol == ')' || symbol == '}'){

if(s.isEmpty() == false){

s.pop();

}

else answer = false;

}

} //Ends for loop

if(!s.isEmpty()){

answer = false;

}

if(answer == true){

return "Valid: even number of brackets";

}

else return "Invalid: odd number of brackets";

}

//Translation

public static String Translate (String eq) {

String translatedString = "";

myQueue q = new myQueue();

intStack s = new intStack();

q.init();

s.init();

char op;

for (int i = 0; i < eq.length(); ++i) {

char[] ch = eq.toCharArray();

if (Character.isLetterOrDigit(ch[i])) {

q.pushQ(ch[i]);

}

else if (ch[i] == '-' || ch[i] == '*' || ch[i] == '+' || ch[i] == '/') {

s.push(ch[i]);

}

else if (ch[i] == ')') {

op = (char)(s.pop());

q.pushQ(op);

}

} //Ends for loop

while (!s.isEmpty()) {

op = (char)(s.pop());

q.pushQ(op);

}

for(int i = 0; i< q. myQueue.length; i++) {

translatedString = translatedString + q.myQueue[i]+ " " ;

}

return translatedString;

}

Console

Shell

line = 73+9*
= 90

line = 52+19**
= 63

line = 89*35+-
= 64

line = 21+84/+
= 5

infix to postfix conversion

line = (8+4)*9
= 8 4 + 9 *

line = (6+3)*(2*8)
= 6 3 + 2 8 * *

line = (9*8)-(5+3)
= 9 8 * 5 3 + -

line = 3*(9/3)
= 3 9 3 / +

Parentheses check, infix to postfix conversion, postfix evaluation

line = (8+4)*9
Valid: even number of brackets
8 4 + 9 *
= 108

line = (6+3)*(2*8)
Valid: even number of brackets
6 3 + 2 8 * *
= 144

line = (9*8)-(5+3)
Valid: even number of brackets
9 8 * 5 3 + -
= 64

line = 3*(9/3)
Valid: even number of brackets
3 9 3 / +
= 6

Bye-bye!
👋

CPU

RAM

Storage

```
1 public class myQueue {  
2  
3     int front;  
4     int rear;  
5  
6     char[] myQueue = new char[10];  
7  
8     public void init() {  
9         front = 0;  
10        rear = -1;  
11    }  
12  
13  
14    public void pushQ(char x) {  
15        rear = rear+1;  
16        myQueue[rear] = x;  
17    }  
18  
19    int popQ() {  
20        char x;  
21        x = myQueue[front];  
22        front = front+1;  
23        return x;  
24    }  
25  
26  
27    public boolean isEmpty() {  
28        boolean empty;  
29        empty = false;  
30        if(front>=rear) {  
31            empty = true;  
32        }  
33        return empty;  
34    }  
35  
36  
37  
38 }
```

```
line = 73+9*  
= 90
```

```
line = 52+19**  
= 63
```

```
line = 89*35+-  
= 64
```

```
line = 21+84/+  
= 5
```

```
*****
```

infix to postfix conversion

```
line = (8+4)*9  
= 8 4 + 9 *
```

```
line = (6+3)*(2*8)  
= 6 3 + 2 8 * *
```

```
line = (9*8)-(5+3)  
= 9 8 * 5 3 + -
```

```
line = 3+(9/3)  
= 3 9 3 / +
```

```
*****
```

Parentheses check, infix to postfix conversion, postfix evaluation

```
line = (8+4)*9  
Valid: even number of brackets  
8 4 + 9 *  
= 108
```

```
line = (6+3)*(2*8)  
Valid: even number of brackets  
6 3 + 2 8 * *  
= 144
```

```
line = (9*8)-(5+3)  
Valid: even number of brackets  
9 8 * 5 3 + -  
= 64
```

```
line = 3+(9/3)  
Valid: even number of brackets  
3 9 3 / +  
= 6
```

Bye-bye!

```
> []
```

Files

AllIn.java

buff.txt

charStack.java

intStack.java

Main.java

myQueue.java

CPU

RAM

Storage

Main.java

```
58 if( line.equals("postfix equation solving")){
59     System.out.println("\n" + line + "\n");
60     line = in.readLine();
61
62 while(!line.equals("infix to postfix conversion")){
63     System.out.println("line = " + line);
64
65     System.out.println("= " + (val.Evaluation(line)));
66     line = in.readLine();
67     System.out.println("\n");
68 } //End While
69
70 //Translation
71
72 System.out.println("*****");
73 if( line.equals("infix to postfix conversion")){
74     System.out.println("\n" + line + "\n");
75     line = in.readLine();
76
77 while(!line.equals("Parentheses check, infix to postfix conversion, postfix evaluation")){
78     System.out.println("line = " + line);
79
80     System.out.println("= " + (val.Translate(line)));
81     line = in.readLine();
82     System.out.println("\n");
83 } //While
84
85 //Translation, Validation, Evaluation
86 System.out.println("*****");
87 if( line.equals("Parentheses check, infix to postfix conversion, postfix evaluation")){
88     System.out.println("\n" + line + "\n");
89     line = in.readLine();
90
91 while(!line.equals("End of file")){
92     System.out.println("line = " + line);
93
94     System.out.println((val.Validation(line)));
95
96 if(val.Validation(line) == "Valid: even number of brackets")
97     System.out.println(val.Translate(line));
98     System.out.println("= " + val.Evaluation(val.Translate(line)));
99     line = in.readLine();
100    System.out.println("\n");
101
102 } //While
103
104 }
105
106 }
107 /* End while. */
108
109 }
110
111 /* End try. */
112
113 catch(IOException e) {
114     System.out.println("Error " + e);
115 } /* End catch. */
```

Console

Shell

line = 73+9*
= 90

line = 52+19**
= 63

line = 89*35+-
= 64

line = 21+84/+
= 5

infix to postfix conversion

line = (8+4)*9
= 8 4 + 9 *

line = (6+3)*(2*8)
= 6 3 + 2 8 * *

line = (9*8)-(5+3)
= 9 8 * 5 3 + -

line = 3+(9/3)
= 3 9 3 / +

Parentheses check, infix to postfix conversion, postfix evaluation

line = (8+4)*9
Valid: even number of brackets
8 4 + 9 *
= 108

line = (6+3)*(2*8)
Valid: even number of brackets
6 3 + 2 8 * *
= 144

line = (9*8)-(5+3)
Valid: even number of brackets
9 8 * 5 3 + -
= 64

line = 3+(9/3)
Valid: even number of brackets
3 9 3 / +
= 6

Bye-bye!
^



```

130 Main.java ×
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121

```

```

Console Shell
line = 73+9*
= 90

line = 52+19**
= 63

line = 89*35+-
= 64

line = 21+84/+
= 5

*****

infix to postfix conversion

line = (8+4)*9
= 8 4 + 9 *

line = (6+3)*(2*8)
= 6 3 + 2 8 * *

line = (9*8)-(5+3)
= 9 8 * 5 3 + -

line = 3+(9/3)
= 3 9 3 / +

*****

Parentheses check, infix to postfix conversion, postfix evaluation

line = (8+4)*9
Valid: even number of brackets
8 4 + 9 *
= 108

line = (6+3)*(2*8)
Valid: even number of brackets
6 3 + 2 8 * *
= 144

line = (9*8)-(5+3)
Valid: even number of brackets
9 8 * 5 3 + -
= 64

line = 3+(9/3)
Valid: even number of brackets
3 9 3 / +
= 6

Bye-bye!

```

Files

AllIn.java

buff.txt

charStack.java

intStack.java

Main.java

myQueue.java

1

public class AllIn{

2

3

//Evaluation

4

int Evaluation(String eq){

5

intStack stacked = new intStack();

6

int num, x, y ,z, j;

7

char myChar;

8

stacked.init();

9

for(j=0; j<eq.length(); j++){

10

myChar = eq.charAt(j);

11

if((myChar >= '0') && (myChar <= '9')){

12

num = myChar - '0';

13

stacked.push(num);

14

}

15

16

else if(myChar == '+'){

17

y = stacked.pop();

18

x = stacked.pop();

19

z = x + y;

20

stacked.push(z);

21

}

22

else if(myChar == '-'){

23

y = stacked.pop();

24

x = stacked.pop();

25

z = x - y;

26

stacked.push(z);

27

}

28

else if(myChar == '/'){

29

y = stacked.pop();

30

x = stacked.pop();

31

z = x / y;

32

stacked.push(z);

33

}

34

else if(myChar == '*'){

35

y = stacked.pop();

36

x = stacked.pop();

37

z = x * y;

38

stacked.push(z);

39

}

40

//Ends for loop

41

42

return stacked.pop();

43

44

}

45

46

//Validation

47

48

String Validation(String eq){

49

int i;

50

charStack s = new charStack();

51

int top;

52

char symbol;

53

s.init();

54

55

boolean answer;

56

answer = true;

57

for(i=0; i<eq.length();i++){

58

symbol = eq.charAt(i);

Console

Shell

line = 73+9*

= 90

line = 52+19**

= 63

line = 89*35+-

= 64

line = 21+84/+

= 5

infix to postfix conversion

line = (8+4)*9

= 8 4 + 9 *

line = (6+3)*(2*8)

= 6 3 + 2 8 * *

line = (9*8)-(5+3)

= 9 8 * 5 3 + -

line = 3+(9/3)

= 3 9 3 / +

Parentheses check, infix to postfix conversion, postfix evaluation

line = (8+4)*9

Valid: even number of brackets

8 4 + 9 *

= 108

line = (6+3)*(2*8)

Valid: even number of brackets

6 3 + 2 8 * *

= 144

line = (9*8)-(5+3)

Valid: even number of brackets

9 8 * 5 3 + -

= 64

line = 3+(9/3)

Valid: even number of brackets

3 9 3 / +

= 6

Bye-bye!

>

CPU

RAM

Storage

sh -c javac -classpath .:target/dependency/ -d . \$(find . -type f -name '*.java')

Note: ./Main.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

✦ java -classpath .:target/dependency/* Main
Hello TV land!
Constructor
Enter file name please.
buff.txt
You entered buff.txt

parentheses tests

$((1+2)*3$

Invalid: odd number of brackets

$(1+2))*3$

Invalid: odd number of brackets

$y=(z*j)/(b*8)+2$

Valid: even number of brackets

postfix equation solving

line = 73+9*
= 90

line = 52+19**
= 63

line = 89*35+-
= 64

line = 21+84/+
= 5

infix to postfix conversion

line = (8+4)*9
= 8 4 + 9 *

line = (6+3)*(2*8)
= 6 3 + 2 8 * *

line = (9*8)-(5+3)
= 9 8 * 5 3 + -

```
line = 52+19**  
= 63
```

```
line = 89*35+-  
= 64
```

```
line = 21+84/+  
= 5
```

infix to postfix conversion

```
line = (8+4)*9
= 8 4 + 9 *
```

```
line = (6+3)*(2*8)
= 6 3 + 2 8 * *
```

```
line = (9*8)-(5+3)
= 9 8 * 5 3 + -
```

```
line = 3+(9/3)
= 3 9 3 / +
```

Parenteses check, infix to postfix conversion, postfix evaluation

```
line = (8+4)*9
Valid: even number of brackets
8 4 + 9 *
= 108
```

```
line = (6+3)*(2*8)
Valid: even number of brackets
6 3 + 2 8 * *
= 144
```

```
line = (9*8)-(5+3)
Valid: even number of brackets
9 8 * 5 3 + -
= 64
```

```
line = 3+(9/3)
Valid: even number of brackets
3 9 3 / +
= 6
```

Bye-bye!

